

Efficiently Finding All Maximal α -gapped Repeats

Paweł Gawrychowski¹, Tomohiro I², Shunsuke Inenaga³, Dominik Köppl², and Florin Manea⁴

- 1 Institute of Informatics, University of Warsaw, Poland,
gawry@mimuw.edu.pl
- 2 Department of Computer Science, TU Dortmund, Germany
{tomohiro.i,dominik.koeppl}@cs.tu-dortmund.de
- 3 Department of Informatics, Kyushu University, Japan
inenaga@inf.kyushu-u.ac.jp
- 4 Department of Computer Science, Kiel University, Germany
flm@informatik.uni-kiel.de

Abstract

For $\alpha \geq 1$, an α -gapped repeat in a word w is a factor uvu of w such that $|uv| \leq \alpha|u|$; the two factors u in such a repeat are called arms, while the factor v is called gap. Such a repeat is called maximal if its arms cannot be extended simultaneously with the same symbol to the right or, respectively, to the left. In this paper we show that the number of maximal α -gapped repeats that may occur in a word is upper bounded by $18\alpha n$. This allows us to construct an algorithm finding all the maximal α -gapped repeats of a word in $\mathcal{O}(\alpha n)$; this is optimal, in the worst case, as there are words that have $\Theta(\alpha n)$ maximal α -gapped repeats. Our techniques can be extended to get comparable results in the case of α -gapped palindromes, i.e., factors uvu^\top with $|uv| \leq \alpha|u|$.

1 Introduction

Gapped repeats and palindromes are repetitive structures occurring in words that were investigated extensively within theoretical computer science (see, e.g., [9, 2, 12, 13, 14, 3, 5, 4, 8, 6, 15] and the references therein) with motivation coming especially from the analysis of DNA and RNA structures, where they were used to model different types of tandem and interspersed repeats as well as hairpin structures; such structures are important in analyzing the structural and functional information of the genetic sequences (see, e.g., [9, 2, 13]).

Following [13, 14], we study gapped repeats (palindromes) uvu (respectively, uvu^\top) where the length of the gap v is upper bounded by the length of the arm u multiplied by some factor, also known as α -gapped repeats and α -gapped palindromes occurring in a word.

The work on α -gapped palindromes was focused so far on combinatorial and algorithmic problems that extend the classical results obtained for squares and palindromes. Namely, problems like how many maximal α -gapped repeats or palindromes does a word of length n contain (here maximal means that the arms of the repeat cannot be both extended to the right or left with the same symbol), how efficiently can we compute the set of maximal α -gapped repeats or palindromes of a word, how efficiently can we compute the longest α -gapped repeat or palindrome, were investigated (see, for instance, [13, 2, 14, 8, 15, 4], and the references therein). In this paper we obtain the following results:

- The number of maximal α -gapped repeats in a word of length n is at most $18\alpha n$.
- We can compute the list of all α -gapped repeats in $\mathcal{O}(\alpha n)$ time for *integer* alphabets.

Our techniques can be extended to show that the number of maximal α -gapped palindromes in a word of length n is upper bounded by $28\alpha n + 7n$; they can be found in $\mathcal{O}(\alpha n)$ time. As



licensed under Creative Commons License CC-BY

Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

there are words of length n that contain $\Theta(\alpha n)$ maximal α -gapped repeats or palindromes (see [14]), it follows that the bounds on the number of maximal α -gapped repeats or palindromes we obtained are asymptotically tight, and that we cannot hope for algorithms finding all α -gapped repeats or palindromes faster in the worst case.

Our results improve those of [14] (as well as those existing in the literature before [14]), where the size of the set of maximal α -gapped repeats with non-empty gap is shown to be $\mathcal{O}(\alpha^2 n)$, and can be computed in $\mathcal{O}(\alpha^2 n)$ time for integer alphabets. An alternative proof of the fact that the number of maximal α -gapped repeats in a word of length n is $\mathcal{O}(\alpha n)$ was given in the very recent [4]; however, compared to the respective paper, we give a more direct proof of the $\mathcal{O}(\alpha n)$ upper bound as well as a concrete evaluation of the constant hidden by the \mathcal{O} -denotation. In [4, 15] algorithms producing all the maximal α -gapped repeats of a word were given; in the light of the upper bound $\mathcal{O}(\alpha n)$ on the number of maximal α -gapped repeats that may occur in a word of length n , it follows that these algorithms work in $\mathcal{O}(\alpha n)$ time, but only for constant alphabets. Extending the approach in [8], we show here that, in fact, such algorithms can be also designed for integer alphabets; our algorithm requires a deeper analysis than the one developed in [8] for finding the longest α -gapped repeat, and uses essentially different techniques and data structures than the ones in [4, 15].

2 Combinatorics on words preliminaries

Let Σ be a finite alphabet; Σ^* denotes the set of all finite words over Σ . The **length** of a word $w \in \Sigma^*$ is denoted by $|w|$. The **empty word** is denoted by ε . A word $u \in \Sigma^*$ is a **factor** of $v \in \Sigma^*$ if $v = xuy$, for some $x, y \in \Sigma^*$; we say that u is a **prefix** of v , if $x = \varepsilon$, and a **suffix** of v , if $y = \varepsilon$. We denote by $w[i]$ the symbol occurring at position i in w , and by $w[i, j]$ the factor of w starting at position i and ending at position j , consisting of the catenation of the symbols $w[i], \dots, w[j]$, where $1 \leq i \leq j \leq n$; we define $w[i, j] = \varepsilon$ if $i > j$. By w^\top we denote the mirror image of w . A **period** of a word w over Σ is a positive integer p such that $w[i] = w[j]$ for all i and j with $i \equiv j \pmod{p}$; a word that has period p is also called **p -periodic**. Let $\text{per}(w)$ be the smallest period of w . A word w with $\text{per}(w) \leq \frac{|w|}{2}$ is called **periodic**; otherwise, w is called **aperiodic**. It is worth noting that the length of the overlap between two consecutive occurrences of an aperiodic factor v in w is upper bounded by $\frac{|v|}{2}$.

By $\mathcal{I} = [b, e]$ we represent the set of consecutive integers from b to e , for $b \leq e$, and call \mathcal{I} an **interval**. For an interval \mathcal{I} , we use the notations $\text{b}(\mathcal{I})$ and $\text{e}(\mathcal{I})$ to denote the beginning and end of \mathcal{I} ; i.e., $\mathcal{I} = [\text{b}(\mathcal{I}), \text{e}(\mathcal{I})]$. We write $|\mathcal{I}|$ to denote the length of \mathcal{I} ; i.e., $|\mathcal{I}| = \text{e}(\mathcal{I}) - \text{b}(\mathcal{I}) + 1$. A **subword** u of a word w is a pair $(s, [b, e])$ consisting of a factor s of w and an interval $[b, e]$ in w such that $s = w[b, e]$. While a factor is identified only by a sequence of letters, a subword is also identified by its position in the word. So subwords are always unique, while a word may contain multiple occurrences of the same factor. For two subwords u and \bar{u} of a word w , we write $u = \bar{u}$ if they start at the same position in w and have the same length. We write $u \equiv \bar{u}$ if the factors identifying these subwords are the same. We implicitly use subwords both like factors of w and as intervals contained in $[1, |w|]$, e.g., we write $u \subseteq \bar{u}$ if two subwords $u = (s, [b, e])$, $\bar{u} = (\bar{s}, [\bar{b}, \bar{e}])$ of w satisfy $[b, e] \subseteq [\bar{b}, \bar{e}]$, i.e., $\text{b}(\bar{u}) \leq \text{b}(u) \leq \text{e}(u) \leq \text{e}(\bar{u})$. Two subwords u and \bar{u} of the same word w are called **consecutive**, iff $\text{e}(u) + 1 = \text{b}(\bar{u})$.

For a word w , we call a triple of consecutive subwords u_λ, v, u_ρ a **gapped repeat** with period $|u_\lambda v|$ and gap $|v|$ iff $u_\rho \equiv u_\lambda$. A triple of consecutive subwords u_λ, v, u_ρ is called a **gapped palindrome** with gap $|v|$ iff $u_\rho \equiv u_\lambda^\top$. The subwords u_λ and u_ρ are called left

and right **arm**, respectively. For $\alpha \geq 1$, the gapped repeat (palindrome) u_λ, v, u_ρ is called **α -gapped** iff $|u_\lambda| + |v| \leq \alpha |u_\lambda|$. Further, it is called **maximal** iff that $w[b(u_\lambda) - 1] \neq w[b(u_\rho) - 1]$ and $w[e(u_\lambda) + 1] \neq w[e(u_\rho) + 1]$, and for a gapped palindrome u_λ, v, u_ρ that $w[b(u_\lambda) - 1] \neq w[e(u_\rho) + 1]$ and $w[e(u_\lambda) + 1] \neq w[b(u_\rho) - 1]$. Let $\mathcal{G}_\alpha(w)$ (respectively, $\mathcal{G}_\alpha^\Gamma(w)$) denote the set of maximal α -gapped repeats (palindromes) in w . The representation of a maximal gapped repeat (palindrome) by the subword $z := w[u_\lambda]w[v]w[u_\rho]$ is not unique — the same subword z can be composed of gapped repeats (palindromes) with different periods (different gaps). Instead, a maximal gapped repeat (palindrome) is uniquely determined by its left arm u_λ and its period (gap). By fixing w , we therefore can map u_λ, v, u_ρ injectively to the pair of integers $(e(u_\lambda), |u_\lambda v|)$ in case of a gapped repeat, or to $(e(u_\lambda), |v|)$ in case of a gapped palindrome.

A run in a word w is a maximal periodic factor; the exponent of a run is the number of times the period fits in that run. For a word w , let $R(w)$ and $E(w)$ denote the number of runs and the sum of the exponents of runs in w , respectively. The exponent of a run r is denoted by $\exp(r)$. We use the following results from literature:

► **Lemma 1** ([1]). *For a word w , $E(w) < 3|w|$.*

► **Lemma 2** ([14]). *Two distinct maximal repetitions with the same minimal period p cannot have an overlap of length greater than or equal to p .*

► **Corollary 3** ([14]). *If a square uu is primitive, any word v contains no more than $|v|/|u|$ occurrences of uu .*

► **Lemma 4.** *Inverting a gapped repeat (palindrome) results in a gapped repeat (palindrome) with the same period. Hence there exist the bijections $\mathcal{G}_\alpha(w) \sim \mathcal{G}_\alpha(w^\Gamma)$ and $\mathcal{G}_\alpha^\Gamma(w) \sim \mathcal{G}_\alpha^\Gamma(w^\Gamma)$.*

3 Point analysis

A pair of positive integers is called a **point**. We use points to bound the cardinality of a subset of gapped repeats and gapped palindromes by injectively mapping a gapped repeat (palindrome) to a point as stated above. To this end, we show that some vicinity of any point generated by a member of this subset does not contain any point that is generated by another member. This vicinity is given by

► **Definition 5.** For any $\gamma \in (0, 1]$, we say that a point (x, y) **γ -covers** a point (x', y') iff $x - \gamma y \leq x' \leq x$ and $y - \gamma y \leq y' \leq y$.

It is crucial that the γ factor is always multiplied with the y -coordinates. In other words, the number of γ -covers of a point (\cdot, y) correlates with γ and the value y . The main property of this definition is given by

► **Lemma 6.** *For any $\gamma \in (0, 1]$, let $S \subset [1, n]^2$ be a set of points such that no two distinct points in S γ -cover the same point. Then $|S| < 3n/\gamma$.*

Proof. We estimate the maximal number of points that can be placed in $[1, n]^2$ such that their covered points are disjoint. First, the number of points $(\cdot, y) \in [1, n]^2$ with $y < 1/\gamma$ is less than n/γ . Second, if a point (\cdot, y) satisfies $2^l/\gamma \leq y < 2^{l+1}/\gamma$ for some integer $l \geq 0$, the point (\cdot, y) γ -covers at least $2^l \times 2^l$ points, or to put it differently, this point γ -covers at least 2^l points (\cdot, y') with $y - 2^l \leq y' \leq y$. In other words, there are at most $n/(2^l \gamma)$ points in S with $2^l/\gamma \leq y < 2^{l+1}/\gamma$. Hence, $|S| < n/\gamma + \sum_{l=0}^{\infty} n/(2^l \gamma) = 3n/\gamma$. ◀

Kolpakov et al. [14] split the set of maximal α -gapped repeats into three subsets, and studied the maximal size of each subset. They analyzed maximal α -gapped repeats by partitioning them into three subsets: those contained in some repetition, those having arms containing a periodic prefix or suffix that is larger than half of the size of the arms, and those not belonging to both former subsets.

They showed that the first two subsets contain at most $\mathcal{O}(\alpha n)$ elements. The point analysis is used as a tool for studying the last subset. By mapping a gapped repeat to a point consisting of the end position of its left arm and its period, they showed that the points created by two different maximal α -gapped repeats cannot $\frac{1}{4\alpha}$ -cover the same point. By this property, they bounded the size of the last subset by $\mathcal{O}(\alpha^2 n)$. Lemma 6 immediately improves this bound of $\mathcal{O}(\alpha^2 n)$ to $\mathcal{O}(\alpha n)$. Consequently, it shows that the number of maximal α -gapped repeats of a word of length n is $\mathcal{O}(\alpha n)$.

4 Gapped repeats

We optimize the proof technique from [14] and improve the upper bound of the number of maximal α -gapped repeats in a word of length n from $\mathcal{O}(\alpha n)$ to $18\alpha n$. Unlike [14, 4], we partition the maximal α -gapped repeats differently. We categorize a gapped repeat (palindrome) depending on whether their left arm contains a periodic prefix or not. Both subsets are treated differently. For the ones having an periodic prefix, we think about the number of runs covering this prefix. The other category is analyzed by using the results of Section 3. We begin with a formal definition of both subsets and analyze the former subset.

Let $0 < \beta < 1$. A gapped repeat (palindrome) $\sigma = u_\lambda, v, u_\rho$ belongs to $\beta\mathcal{P}_\alpha(w)$ ($\beta\mathcal{P}_\alpha^\Gamma(w)$) iff u_λ contains a periodic prefix of length at least $\beta|u_\lambda|$. We call σ **periodic**. Otherwise $\sigma \in \bar{\beta\mathcal{P}}_\alpha(w)$ ($\sigma \in \bar{\beta\mathcal{P}}_\alpha^\Gamma(w)$), where $\bar{\beta\mathcal{P}}_\alpha(w) := \mathcal{G}_\alpha(w) \setminus \beta\mathcal{P}_\alpha(w)$ and $\bar{\beta\mathcal{P}}_\alpha^\Gamma(w) := \mathcal{G}_\alpha^\Gamma(w) \setminus \beta\mathcal{P}_\alpha^\Gamma(w)$; we call σ **aperiodic**.

► **Lemma 7.** *Let w be a word, $\alpha > 1$ and $0 < \beta < 1$ two real numbers. Then $|\beta\mathcal{P}_\alpha(w)|$ is at most $2\alpha E(w)/\beta$.*

Proof. Let $\sigma = (u_\lambda, v, u_\rho) \in \beta\mathcal{P}_\alpha(w)$. By definition, the left arm u_λ has a periodic prefix s_λ of length at least $\beta|u_\lambda|$. Let r_λ denote the run that generates s_λ , i.e., $s_\lambda \subseteq r_\lambda$ and they both have the common shortest period p . By the definition of gapped repeats, there is a right copy s_ρ of s_λ contained in u_ρ with $s_\rho = w[b(s_\lambda) + |u_\lambda v|, e(s_\lambda) + |u_\lambda v|] \equiv s_\lambda$.

Let r_ρ be a run generating s_ρ (it is possible that r_ρ and r_λ are identical). By definition, r_ρ has the same period p as r_λ . In the following, we will see that σ is uniquely determined by r_λ and the period $q := |u_\lambda v|$, if σ is a periodic gapped repeat. We will fix r_λ and pose the question how many maximal periodic gapped repeats can be generated by r_λ .

Since σ is maximal, $b(u_\lambda) = b(r_\lambda)$ or $b(u_\rho) = b(r_\rho)$ must hold; otherwise we could extend σ to the left. We analyze the case $b(s_\lambda) = b(r_\lambda)$, the other is treated exactly in the same way by symmetry. The gapped repeat σ is identified by r_λ and the period q . We fix r_λ and count the number of possible values for the period q . Given two different gapped repeats σ_1 and σ_2 with respective periods q_1 and q_2 such that the left arms of both are generated by r_λ , the difference δ between q_1 and q_2 must be at least p .

Since $|u_\lambda| \leq |s_\lambda|/\beta$ and σ is α -gapped, $1 \leq q \leq |s_\lambda|\alpha/\beta \leq |r_\lambda|\alpha/\beta$. Then the number of possible periods q is bounded by $|r_\lambda|\alpha/(\beta p) = \exp(r_\lambda)\alpha/\beta$. Therefore the number of maximal α -gapped repeats is bounded by $\alpha E(w)/\beta$ for the case $b(u_\lambda) = b(r_\lambda)$. Summing up we get the bound $2\alpha E(w)/\beta$. ◀

Remembering the results of Section 3, we map gapped repeats to their respective points. By using the period as the y -coordinate, one can show Lemma 8.

► **Lemma 8.** *Given a word w , and two real numbers $\alpha > 1$ and $2/3 \leq \beta < 1$. The points mapped by two different maximal gapped repeats in $\beta\mathcal{P}_\alpha(w)$ cannot $\frac{1-\beta}{\alpha}$ -cover the same point.*

Proof. Let $\sigma = u_\lambda v u_\rho$ and $\bar{\sigma} = \bar{u}_\lambda \bar{v} \bar{u}_\rho$ be two different maximal gapped repeats in $\beta\mathcal{P}_\alpha(w)$. Set $u := |u_\lambda| = |u_\rho|$, $\bar{u} := |\bar{u}_\lambda| = |\bar{u}_\rho|$, $q := |u_\lambda v|$ and $\bar{q} := |\bar{u}_\lambda \bar{v}|$. We map the maximal gapped repeats σ and $\bar{\sigma}$ to the points $(e(u_\lambda), q)$ and $(e(\bar{u}_\lambda), \bar{q})$, respectively. Assume, for the sake of contradiction, that both points $\frac{1-\beta}{\alpha}$ -cover the same point (x, y) .

Let $z := |e(u_\lambda) - e(\bar{u}_\lambda)|$ be the difference of the endings of both left arms, and $s_\lambda := w[b(u_\lambda), e(u_\lambda)] \cap [b(\bar{u}_\lambda), e(\bar{u}_\lambda)]$ be the overlap of u_λ and \bar{u}_λ . Let $s := |s_\lambda|$, and let s_ρ (resp. \bar{s}_ρ) be the right copy of s_λ based on σ (resp. $\bar{\sigma}$).

Sub-Claim: The overlap s_λ is not empty, and $s_\rho \neq \bar{s}_\rho$

Sub-Proof. Assume for this sub-proof that $e(u_\lambda) < e(\bar{u}_\lambda)$ (otherwise exchange σ with $\bar{\sigma}$, or yield the contradiction $\sigma = \bar{\sigma}$). By combining the $(1-\beta)/\alpha$ -cover property with the fact that $\bar{\sigma}$ is α -gapped, we yield $e(\bar{u}_\lambda) - \bar{u} \leq e(\bar{u}_\lambda) - \bar{q}(1-\beta)/\alpha \leq x \leq e(u_\lambda) < e(\bar{u}_\lambda)$. So the subword $w[e(u_\lambda)]$ is contained in \bar{u}_λ . If $s_\rho = \bar{s}_\rho$, then we get a contradiction to the maximality of σ : By the above inequality, $w[e(u_\lambda) + 1]$ is contained in \bar{u}_λ , too. Since $\bar{\sigma}$ is a gapped repeat, the character $w[e(u_\lambda) + 1]$ occurs in \bar{u}_ρ , exactly at $w[e(u_\rho) + 1]$. ◀

Without loss of generality let $q \leq \bar{q}$. Then

$$\bar{q} - \frac{\bar{q}(1-\beta)}{\alpha} \leq y \leq q \leq \bar{q}. \quad (1)$$

$$\text{So the difference of both periods is } 0 \leq \delta := \bar{q} - q \leq \bar{q}(1-\beta)/\alpha \leq \bar{u}(1-\beta). \quad (2)$$

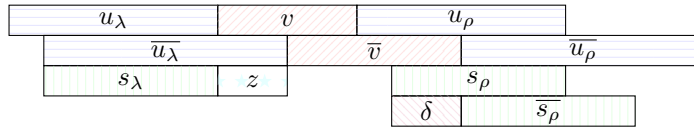
$$\text{Equation (1) also yields that } u \geq q/\alpha \geq \frac{\bar{q}}{\alpha}(1 - \frac{1-\beta}{\alpha}) \geq \bar{q}\beta/\alpha. \quad (3)$$

Since $s_\rho = [b(s_\lambda) + q, e(s_\lambda) + q]$ and $\bar{s}_\rho = [b(s_\lambda) + \bar{q}, e(s_\lambda) + \bar{q}]$, we have $b(\bar{s}_\rho) - b(s_\rho) = \delta$.

By case analysis, we show that u_λ or \bar{u}_λ has a periodic prefix, which leads to the contradiction that σ or $\bar{\sigma}$ are in $\beta\mathcal{P}_\alpha(w)$.

1. Case $e(u_\lambda) \leq e(\bar{u}_\lambda)$. Since $e(\bar{u}_\lambda) - \bar{q}(1-\beta)/\alpha \leq x \leq e(u_\lambda) \leq e(\bar{u}_\lambda)$,

$$z = e(\bar{u}_\lambda) - e(u_\lambda) \leq \bar{q}(1-\beta)/\alpha \leq \bar{u}(1-\beta). \quad (4)$$



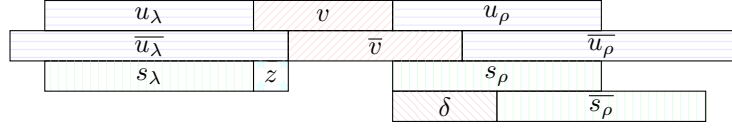
■ **Figure 1** Sub-Case 1a

1a. Sub-Case $b(u_\lambda) \leq b(\bar{u}_\lambda)$. By Equation (4), we get $s = \bar{u} - z \geq \bar{u}\beta$. It follows from Equation (2) and $2/3 \leq \beta < 1$ that $s/\delta \geq \bar{u}\beta/\bar{u}(1-\beta) = \beta/(1-\beta) \geq 2$, which means that s_ρ and \bar{s}_ρ overlap at least half of their common length, so s_λ is periodic. Since s_λ is a prefix of \bar{u}_λ of length $s \geq \bar{u}\beta$, $\bar{\sigma}$ is in $\beta\mathcal{P}_\alpha(w)$, a contradiction.

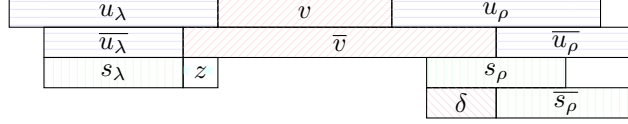
1b. Sub-Case $b(u_\lambda) > b(\bar{u}_\lambda)$. We conclude that $s_\lambda = u_\lambda$. It follows from Equations (2) and (3) and $2/3 \leq \beta < 1$ that $s/\delta \geq \bar{q}\alpha\beta/(\bar{q}\alpha(1-\beta)) = \beta/(1-\beta) \geq 2$, which means that $s_\lambda = u_\lambda$ is periodic. Hence σ is in $\beta\mathcal{P}_\alpha(w)$, a contradiction.

2. Case $e(u_\lambda) > e(\bar{u}_\lambda)$. Since $e(u_\lambda) - q(1-\beta)/\alpha \leq x \leq e(\bar{u}_\lambda) \leq e(u_\lambda)$,

$$z = e(u_\lambda) - e(\bar{u}_\lambda) \leq q(1-\beta)/\alpha \leq \bar{q}(1-\beta)/\alpha \leq \bar{u}(1-\beta). \quad (5)$$

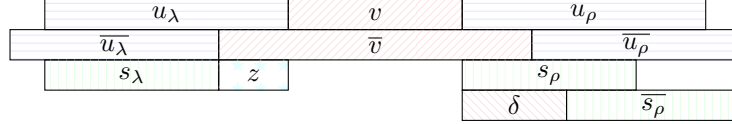


■ **Figure 2** Sub-Case 1b



■ **Figure 3** Sub-Case 2a

2a. Sub-Case $b(u_\lambda) \leq b(\overline{u_\lambda})$. We conclude that $s_\lambda = \overline{u_\lambda}$. It follows from Equation (2) and $2/3 \leq \beta < 1$ that $s/\delta \geq \overline{u}/(\overline{u}(1-\beta)) = 1/(1-\beta) \geq 3 > 2$, which means that $s_\lambda = \overline{u_\lambda}$ is periodic. Hence $\overline{\sigma}$ is in $\beta\mathcal{P}_\alpha(w)$, a contradiction.



■ **Figure 4** Sub-Case 2b

2b. Sub-Case $b(u_\lambda) > b(\overline{u_\lambda})$. By Equation (5), we get $s = u - z \geq u\beta$. If $\delta \leq s/2$, s_ρ and $\overline{s_\rho}$ overlap at least half of their common length, which leads to the contradiction that u_λ has a periodic prefix s_λ of length at least $u\beta$. Otherwise, let us assume that $s/2 < \delta$. By Equations (2) and (3) we get $u/\delta \geq \overline{q}\alpha\beta/(\overline{q}\alpha(1-\beta)) = \beta/(1-\beta) \geq 2$ with $2/3 \leq \beta < 1$. Hence, δ is upper bounded by $u/2$; so u_ρ has a periodic prefix of length at least 2δ (since $2\delta > s \geq u\beta$), a contradiction. ◀

The next lemma follows immediately from Lemmas 6 and 8.

► **Lemma 9.** For $\alpha > 1$, $2/3 \leq \beta < 1$ and a word w of length n , $|\overline{\beta\mathcal{P}_\alpha}(w)| < 3\alpha n/(1-\beta)$.

► **Theorem 10.** Given a word w of length n , and a real number $\alpha > 1$. Then $|\mathcal{G}_\alpha(w)| < 18\alpha n$.

Proof. Combining the results of Lemmas 7 and 9, $|\mathcal{G}_\alpha(w)| = |\beta\mathcal{P}_\alpha(w)| + |\overline{\beta\mathcal{P}_\alpha}(w)| < 2\alpha E(w)/\beta + 3\alpha n/(1-\beta)$ for $2/3 \leq \beta < 1$. Applying Lemma 1, the term is upper bounded by $6\alpha n/\beta + 3\alpha n/(1-\beta)$. The number is minimal for $\beta = 2/3$, yielding the bound $18\alpha n$. ◀

We can bound the number of maximal α -gapped palindromes by similar proofs to $28\alpha n + 7n$. This bound solves an open problem in [13], where Kolpakov and Kucherov conjectured that the number of α -gapped palindromes with $\alpha \geq 2$ in a string is linear. We briefly explain the main differences and similarities needed to understand the relationship between gapped repeats and palindromes. Let σ be a maximal α -gapped repeat (or α -gapped palindrome). If the gapped repeat (palindrome) has a periodic prefix s_λ generated by some run, the right arm has a periodic prefix (suffix) s_ρ generated by a run of the same period. Since σ is maximal, both runs have to obey constraints that are similar in both cases, considering whether σ is a gapped repeat or a gapped palindrome. So it is easy to change the proof of Lemma 7 in order to work with palindromes. Like with aperiodic gapped repeats, we

can apply the point analysis to the aperiodic α -gapped palindromes, too. As main idea, we map a gapped palindrome u_λ, v, u_ρ injectively to the pair of integers $(e(u_\lambda), |v|)$, exchanging the period with the size of the gap. See appendix for proofs.

5 Algorithms

The computational model we use to design and analyze our algorithms is the standard unit-cost RAM with logarithmic word size, which is generally used in the analysis of algorithms. In the upcoming algorithmic problems, we assume that the words we process are sequences of integers. In general, if the input word has length n then we assume its letters are in $\{1, \dots, n\}$, so each letter fits in a single memory-word. This is a common assumption in stringology (see, e.g., the discussion in [10]). For a word w , $|w| = n$, we build in $\mathcal{O}(n)$ time the suffix array as well as data structures allowing us to retrieve in constant time the length of the longest common prefix of any two suffixes $w[i, n]$ and $w[j, n]$ of w , denoted $LCP_w(i, j)$ (the subscript w is omitted when there is no danger of confusion). Such structures are called *LCP* data structures in the following (see, e.g., [10, 9]). We begin with a simple lemma.

► **Lemma 11.** *Given a word w , $|w| = n$, we can process it in $\mathcal{O}(n)$ time such that, for each $i, p \leq n$, we can return in $\mathcal{O}(1)$ time the longest factor of period p starting at position i in w .*

Let w be a word and v be a factor of w with $\text{per}(v) = p$. Further, let z be a subword of length $\ell|v|$ of w . An occurrence of v in z is a subword $(v, [i, i + |v| - 1])$ of z ; we say that v occurs at position i in z . For an easier presentation of our algorithm, we distinguish between two types of occurrences of v in z . On the one hand, we have the so-called **single occurrences**. If v is aperiodic, then all its occurrences in z are single occurrences; there are $\mathcal{O}(\ell)$ such occurrences (see, e.g., [11]). If v is periodic, then a subword $(v, [i, i + |v| - 1])$ of z starting on position i in z is a single occurrence if v occurs neither at position $i - p$ nor at position $i + p$ in z . On the other hand, we have **occurrences of v within a run** of z , whose period is $p = \text{per}(v)$. That is, the subword $(v, [i, i + |v| - 1])$ starting on position i in z is an occurrence of v within a run if v occurs either at $i - p$ or at $i + p$. We say that $(v, [i, i + |v| - 1])$ is the **first occurrence** of v in a run of period p of z if v does not occur at $i - p$ but occurs at $i + p$. Note that there are $\mathcal{O}(\ell)$ runs containing occurrences of v in z , or, equivalently, $\mathcal{O}(\ell)$ first occurrences of v in a run of period p .

Consequently, the occurrences of v in z can be succinctly represented as follows. For the single occurrences we just store their starting position. The occurrences of v in a run r can be represented by the starting position of the first occurrence of v in r , together with the period of v , since the starting positions of the occurrences of v in r form an arithmetic progression of period p .

In our approach, basic factors (i.e., factors of length 2^k , for $k \geq 1$) of the input word are important. For some integer $c \geq 2$, the occurrences of the basic factor $w[i, i + 2^k - 1]$ in a subword of length $c2^k$ can be represented in a compact manner: $\mathcal{O}(c)$ positions of the single occurrences of $w[i, i + 2^k - 1]$ and $\mathcal{O}(c)$ first occurrences of $w[i, i + 2^k - 1]$ in runs, together with the period of $w[i, i + 2^k - 1]$. We recall the next lemma (see [8, 11], appendix).

► **Lemma 12.** *Given a word w of length n and an integer $c \geq 2$, we can process w in time $\mathcal{O}(n \log n)$ such that given any basic factor $y = w[i, i + 2^k - 1]$ and any subword of w $(z, [j, j + c2^k - 1])$, with $k \geq 0$, we can compute in $\mathcal{O}(\log \log n + c)$ time the representation of all the (single and within runs) occurrences of y in z .*

We now focus on short basic factors of words. The constant 16 occurring in the following considerations can be replaced by any other constant; we just use it here so that we can

apply these results directly in the main proofs of this section.

Given a word v and some integer $\beta \geq 16$ with $|v| = \beta \log n$, as well as a basic factor $y = v[i2^k + 1, (i+1)2^k]$, with $i, k \geq 0$ and $i2^k + 1 > (\beta - 16) \log n$ (so occurring in the suffix of length $16 \log n$ of v), the occurrences of y in v can be represented as $\mathcal{O}(\beta)$ bit-sets, each containing $\mathcal{O}(\log n)$ bits, the 1-bits marking the starting positions of the occurrences of y in v . The next result can be shown using tools developed in [7] (see also [8] and the appendix).

► **Lemma 13.** *Given a word v and an integer $\beta > 16$, with $|v| = \beta \log n$, we can process v in time $\mathcal{O}(\beta \log n)$ time such that given any basic factor $y = v[i2^k + 1, (i+1)2^k]$ with $i, k \geq 0$ and $i2^k + 1 > (\beta - 16) \log n$, we can find in $\mathcal{O}(\beta)$ time the $\mathcal{O}(\beta)$ bit-sets, each storing $\mathcal{O}(\log n)$ bits, characterizing all the occurrences of y in v .*

In the context of the previous lemma, once the occurrences of y in v are computed, given a subword z of v of length $|z| = c|y|$, for some $c \geq 1$, we can obtain in $\mathcal{O}(c)$ time both the single occurrences of y in z and the occurrences of y within runs of z . We just have to select (by bitwise operations on the bit-sets encoding the factors of v that overlap z) the positions where y occurs (so the positions of the 1-bits in those bit-sets). For each two consecutive such occurrences of y we detect whether they are part of a run in v and then skip over all the occurrences of y from that run (and the corresponding parts of the bit-sets) before looking again for 1-bits in the bit-sets; for the positions that form a run we store the first occurrence of y and its period, while for the single occurrences we store the position of that occurrence.

Now we can begin the presentation of the algorithm finding all the maximal α -gapped repeats of a word. We first show how to find maximal repeats with short arms.

► **Lemma 14.** *Given a word w and $\alpha \geq 1$, we can find all the maximal α -gapped repeats u_λ, u', u_ρ occurring in w , with $|u_\rho| \leq 16 \log n$, in time $\mathcal{O}(\alpha n)$.*

Proof. If a maximal α -gapped repeat u_λ, u', u_ρ (where we denote by u the underlying factor of both arms), has $|u| \leq 16 \log n$, we get that u_ρ must be completely contained in a subword $(w', [m \log n + 1, (m+17) \log n])$, for some m with $\frac{n}{\log n} - 17 \geq m \geq 0$. By fixing the interval where u_ρ may occur (that is, fix m), we also fix the place where u_λ may occur. Indeed, the entire subword u_λ, u', u_ρ is completely contained in the factor $x_m = (w'', [(m-16\alpha) \log n + 1, (m+17) \log n])$ (or, in a factor $x_m = (w'', [1, (m+17) \log n])$ if $(m-16\alpha) \log n + 1 < 1$).

Hence, we look for maximal α -gapped repeats u_λ, u', u_ρ completely contained in x_m with u_ρ completely contained in the suffix of length $16 \log n$ of x_m ; then we repeat this process for all m . To begin with, we process x_m as in Lemma 13, and construct *LCP*-structures for it.

Now, once we fixed the subword x_m of w where we search the maximal α -gapped repeats, we try to fix also their length. That is, we find all maximal α -gapped repeats u_λ, u', u_ρ with $2^{k+1} \leq |u| \leq 2^{k+2}$ completely contained in x_m with u_ρ completely contained in the suffix of length $16 \log n$ of x_m ; we execute this process for all $0 \leq k \leq \log(16 \log n)$. Note that all the maximal α -gapped repeats with arms shorter than 2 (occurring anywhere in the word w) can be trivially found in $\mathcal{O}(\alpha n)$ time.

Since we can find occurrences of basic factors in x_m efficiently, we try to build a maximal gapped repeat by extending gapped repeats whose arms contain a basic factor (see Figure 12 in the appendix). To this end, we analyze some *subwords* of x_m : If $2^{k+1} \leq |u| \leq 2^{k+2}$ then u_ρ contains at least one subword $(y, [j2^k + 1, (j+1)2^k])$ starting within its first 2^k positions. A copy of the factor y occurs also within the first 2^k positions of u_λ (with the same offset with respect to the starting position of u_λ as the offset of the occurrence of y with respect to the starting position of u_ρ). So, finding the respective copy of y from u_λ helps us discover

the place where u_λ actually occurs. Indeed, assume that we identified the copy of y from u_λ , and assume that this copy is $(y, [\ell + 1, \ell + |y|])$; we try to build u_λ and u_ρ around these two occurrences of y , respectively. Hence, in order to identify u_λ and u_ρ we compute the longest factor p of x_m that ends both on $j2^k$ and on ℓ and the longest factor s that starts both on $(j + 1)2^k + 1$ and on $\ell + |y| + 1$. Now, if $\ell + |y| + |s| \leq j2^k - |p|$ then u_λ is obtained by concatenating p and s around $x_m[\ell + 1, \ell + |y|]$ while u_ρ is obtained by concatenating p and s to the left and, respectively, right of $x_m[j2^k + 1, (j + 1)2^k]$; otherwise, the two occurrences of y do not determine a maximal repeat. Moreover, the repeat we determined is a valid solution of our problem only if its length is between 2^{k+1} and 2^{k+2} , and its right arm contains position $j2^k + 1$ of x_m within its first 2^k positions.

Now we explain how to determine efficiently the copy of y around which we try to build u_λ . As $|u| < 2^{k+2}$ and $|y| = 2^k$ we get that the copy of y that corresponds to u_λ should be completely contained in the subword of x_m of length $\alpha 2^{k+2}$ ending on position $j2^k$. As said above, we already processed x_m to construct the data structures from Lemma 13. Therefore, we can obtain in $\mathcal{O}(\alpha)$ time a representation of all the occurrences of y inside the factor of length $\alpha 2^{k+2}$ ending on position $j2^k$. These occurrences can be single occurrences and occurrences within runs. There are $\mathcal{O}(\alpha)$ single occurrences, and we can process each of them individually, as explained, to find the maximal α -gapped repeat they determine together with the occurrence of y from u_ρ . However, it is not efficient to do the same for the occurrences of y within runs. For these (which are also $\mathcal{O}(\alpha)$ many) we proceed as follows.

Assume we have a run of occurrences of y inside the factor of x_m of length $\alpha 2^{k+2}$ ending on position $j2^k$. Let ℓ be the starting position of the first occurrence of y in this run and let p be the period of y . Now, using Lemma 11 we can determine the maximal p -periodic subword r_λ of x_m containing this run of y -occurrences. Similarly, we can determine the maximal p -periodic subword r_ρ that contains the occurrence of y from u_ρ (i.e., $x_m[j2^k + 1, (j + 1)2^k]$). To determine efficiently the α -gapped repeats that contain $x_m[j2^k + 1, (j + 1)2^k]$ in the right arm and a corresponding occurrence of y from r_λ in the left arm we analyze several cases (see Figure 13 in the appendix).

Assume u_ρ starts on a position of r_ρ , other than its first one. Then u_λ should also start on the first position of r_λ (or we could extend both arms to the left, a contradiction to the maximality of the repeat). If u_ρ ends on a position to the right of r_ρ , then u_λ also ends on a position to the right of r_λ , and, moreover, the suffix of u_λ occurring after the end of r_λ and the suffix of u_ρ occurring after the end of r_ρ are equal, and can be computed by a longest common prefix query on x_m . This means that u_λ can be determined exactly (we know where it starts and where it ends) so u_ρ can also be determined exactly (we know where it ends), and we can check if the obtained repeat is indeed a maximal α -gapped repeat, and the arms fulfill the required length conditions (i.e., length between 2^{k+1} and 2^{k+2} , the right arm contains position $j2^k + 1$ of x_m within its first 2^k positions). If u_ρ ends exactly on the same position as r_ρ then u_ρ is periodic of period p ; we just have to compute the longest p -periodic factor that ends on the same position as r_ρ and starts at the same position as r_λ , and this can also be determined in constant time just by taking the longest p -periodic prefix of r_λ which is also a suffix of r_ρ . So, again, we can determine exactly u_λ and u_ρ , and we can check if they form a maximal α -gapped repeat with the arms fulfilling the length restrictions. The final, and more complicated case, is when u_ρ ends on a position of r_ρ , other than its last position. In that case, we get that $u_\lambda = r_\lambda$ (or, otherwise, we could extend both arms to the right). Essentially, this means that we know exactly where u_λ is located and its length (and we continue only if this length is between 2^{k+1} and 2^{k+2}); so u_λ denotes a factor $z^h z'$ for some z of length p . Now, looking at the run r_ρ , we can get easily the position of the first

occurrence of z in that run, and the position of its last occurrence. If the first occurrence is ℓ' , then the occurrences of z have their starting positions $\ell', \ell' + p, \dots, \ell' + tp$ for some t . As we know the length of u_λ and the fact that u_λ, u', u_ρ is α -gapped, we can determine in constant time the values $0 \leq i \leq t$ such that u_ρ may start on position $\ell' + ip$, the repeat we obtain is α -gapped, and u_ρ contains position $j2^k + 1$ of x_m within its first 2^k positions. If u_ρ is a prefix of r_ρ we also have to check that we cannot extend simultaneously u_ρ and u_λ to the left; if u_ρ is a suffix of r_λ we have to check that we cannot extend simultaneously u_ρ and u_λ to the right. Then we can return the maximal α -gapped repeats we constructed.

The cases when u_ρ starts on the first position of r_ρ or when it starts on a position to the left of r_ρ can be treated similarly, and as efficiently (see Appendix).

This concludes our algorithm. Its correctness follows from the explanations above. Moreover, we can ensure that our algorithm finds and outputs each maximal repeat exactly once; this clearly holds when we analyze the repeats of x_m for each m separately. However, when moving from x_m to x_{m+1} we must also check that the right arm of each repeat we find is not completely contained in x_m (so, already found). This condition can be easily imposed in our search: when constructing the arms determined by a single occurrence of y , we check the containment condition separately; when constructing a repeat determined by a run of y -occurrences, we have to impose the condition that the right arm extends out of x_m when searching the starting positions of the possible arms.

Next, we compute the complexity of the algorithm. Once we fix m, k , and j , our process takes $\mathcal{O}(\alpha + N_{j,m,k})$ time, where $N_{j,m,k}$ is the number of maximal α -gapped repeats determined for the fixed m, j, k . So, the time complexity of the algorithm is:

$\mathcal{O}(n + \sum_{0 \leq m \leq n/\log n} (16\alpha \log n + \sum_{0 \leq k \leq \log(16 \log n)} (\sum_{j \leq 16 \log n/2^k} (\alpha + N_{j,m,k})))) \subseteq \mathcal{O}(\alpha n)$, as the total number of maximal α -gapped repeats is $\mathcal{O}(\alpha n)$ and we need $\mathcal{O}(|x_m|)$ preprocessing time for each x_m and $\mathcal{O}(n)$ preprocessing time for w . \blacktriangleleft

Next, we find all maximal α -gapped repeats with longer arms.

► **Lemma 15.** *Given a word w and $\alpha \geq 1$, we can find all the maximal α -gapped repeats u_λ, u', u_ρ occurring in w , with $|u_\rho| > 16 \log n$, in time $\mathcal{O}(\alpha n)$.*

Proof. The general approach in proving this lemma is similar to that used in the proof of the previous result. Essentially, when identifying a new maximal α -gapped repeat, we try to fix the place and length of the right arm u_ρ of the respective repeat, which restricts the place where the left arm u_λ occurs. This allows us to fix some long enough subword of w as being part of the right arm, detect its occurrences that are possibly contained in the left arm, and, finally, to efficiently identify the actual repeat. The main difference is that we cannot use the result of Lemma 13, as we have to deal with repeats with arms longer than $16 \log n$. Instead, we will use the structures constructed in Lemma 12. However, to get the stated complexity, we cannot apply this lemma directly on the word w , but rather on an encoded variant of w .

Thus, the first step of the algorithm is to construct a word w' , of length $\frac{n}{\log n}$, whose symbols, called **blocks**, encode $\log n$ consecutive symbols of w grouped together. That is, the first block of the new word corresponds to $w[1, \log n]$, the second one to $w[\log n + 1, 2 \log n]$, and so on. Hence, we have two versions of the word w : the original one, and the one where it is split in blocks. It is not hard to see that the blocks can be encoded into numbers between 1 and n in linear time. Indeed, we build the suffix array and *LCP*-data structures for w , and then we cluster together the suffixes of the suffix array that share a common prefix of length at least $\log n$. Then, all the suffixes of a cluster are given the same number (between 1 and n), and a block is given the number of the suffix starting with the respective block.

We can now construct in $\mathcal{O}(n)$ time the suffix arrays and *LCP*-data structures for both w and w' , as well as the data structures of Lemma 12 for the word w' .

Now, we guess the length of the arms of the repeat. We try to find the maximal α -gapped repeats u_λ, u', u_ρ of w with $2^{k+1} \log n \leq |u_\lambda| \leq 2^{k+2} \log n$, $k \leq \log \frac{n}{\log n} - 2$. We fix k and split again the word w , this time in factors of length $2^k \log n$, called *k-blocks*. Assume that each split is exact (padding the word with some new symbols ensures this).

Now, if a maximal α -gapped repeat u_λ, u', u_ρ with $2^{k+1} \log n \leq |u_\lambda| \leq 2^{k+2} \log n$ exists, then it contains an occurrence of a k -block within its first $2^k \log n$ positions. So, let z be a k -block and assume that it is the first k -block occurring in u_ρ (in this way fixing a range where u_ρ may occur). Obviously, if u_ρ contains z , then u_λ also contains an occurrence of z ; however, this occurrence is not necessarily starting on a position $j \log n + 1$ for some $j \geq 0$ (so, it is not necessarily a sequence of blocks). But, at least one of the factors of length 2^{k-1} starting within the first $\log n$ positions of z (which are not necessarily sequences of blocks) must correspond, in fact, to a sequence of blocks from the left arm u_λ . So, let us fix now a factor y of length 2^{k-1} that starts within the first $\log n$ positions of z (we try all of them in the algorithm, one by one). As said, the respective occurrence of y from u_ρ is not necessarily a sequence of blocks (so it cannot be mapped directly to a factor of w'). But, we look for an occurrence of y starting on one of the $\alpha 2^{k+2} \log n$ positions to the left of z , corresponding to a sequence of blocks, and assume that the respective occurrence is exactly the occurrence of y from u_λ .

By binary searching the suffix array of w' (using *LCP*-queries on w to compare the factors of $\log n$ symbols of y and the blocks of w' , at each step of the search) we try to detect a factor of w' that encodes a word equal to y . Assume that we can find such a sequence y' of 2^{k-1} blocks of w' (otherwise, y cannot correspond to a sequence of blocks from u_λ , so we should try other factors of z instead). Using Lemma 12 for w' , we get in $\mathcal{O}(\log \log |w'| + \alpha)$ time a representation of the occurrences of y' in the range of $\alpha 2^{k+2}$ blocks of w' occurring before the blocks of z ; this range corresponds to an interval of w with a length of $\alpha 2^{k+2} \log n$.

Further, we process these occurrences of y' just like in the previous lemma. Namely, the occurrences of y' in that range are either single occurrences or occurrences within runs. Looking at their corresponding factors from w , we note that each of these factors fixes a possible left arm u_λ ; this arm, together with the corresponding arm u_ρ can be constructed just like before. In the case of single occurrences (which are at most $\mathcal{O}(\alpha)$, again), we try to extend both the respective occurrence and the occurrence of y from u_ρ both to the left and, respectively, to the right, simultaneously, and see if we can obtain in this way the arms of a valid maximal α -gapped repeat. Note that we must check also that the length of the arm of the repeat is between 2^{k+1} and 2^{k+2} , and that z is the first k -block of the right arm. As before, complications occur when the occurrences of y' are within runs. In this case, the run of occurrences of y' does not necessarily give us the period of y , but a multiple of this period that can be expressed also as a multiple of $\log n$ (or, in other words, the minimum period of y is a multiple of the block-length). This, however, does not cause any problems, as the factor y from u_ρ should always correspond to a block sequence from u_λ , so definitely to one of the factors encoded in the run of occurrences of y' .

Therefore, by determining the maximal factor that contains y and has the same period as the run of occurrences of y' (with the period measured in w), we can perform a very similar analysis to the corresponding one from the case when we searched maximal α -gapped repeats with arms shorter than $16 \log n$.

It remains to prove that each maximal gapped repeat is counted only once. Essentially, the reason for this is that for two separate factors y_1 and y_2 (of length 2^{k-1}) occurring in the

first $\log n$ symbols of z we cannot get occurrences of the corresponding factors y'_1 and y'_2 that define the same repeat; in that case, the distance between y'_1 and y'_2 should be at least one block, so the distance between y_1 and y_2 should be at least $\log n$, a contradiction. Similarly, if we have a factor y occurring in the first $\log n$ symbols of some k -block z_1 such that this factor determines an α -gapped maximal repeat, then the same maximal repeat cannot be determined by a factor of another k -block, since z_1 is the first k -block of u_ρ .

The correctness of the algorithm described above follows easily from the explanations given in the proofs of the last two lemmas. Let us evaluate its complexity. The preprocessing phase (construction of w' and of all the needed data structures) takes $\mathcal{O}(n)$ time. Further, we can choose k (and implicitly an interval for the length of the arms of the repeats) such that $k \leq \log \frac{n}{\log n} - 2$. After choosing k , we can choose a k -block z in $\frac{n}{2^k \log n}$ ways. Further, we analyze each factor y of length 2^{k-1} starting within the first $\log n$ positions of the chosen k -block z . For each such factor y we find in $\mathcal{O}(\log \frac{n}{\log n} + \log \log n + \alpha)$ time the representation of the occurrences of the block encoding the occurrence of y from u_λ . From each of the $\mathcal{O}(\alpha)$ single occurrences we check whether it is possible to construct a maximal α -gapped repeat in $\mathcal{O}(1)$ time. We also have $\mathcal{O}(\alpha)$ occurrences of the block encoding y in runs, and each of them is processed in $\mathcal{O}(N_{z,y})$ time, where $N_{z,y}$ is the number of maximal α -gapped repeats we find for some z and y . Overall, this adds up to a total time of $\mathcal{O}(n \log n + \alpha n)$, as the total number of maximal α -gapped repeats in w is upper bounded by $\mathcal{O}(\alpha n)$. If $\alpha \geq \log n$, the statement of the lemma follows. If $\alpha < \log n$ we proceed as follows.

Initially, we run the algorithm only for $k > \log \log n$ and find the maximal α -gapped repeats $u_\lambda u' u_\rho$ with $2^{\log \log n} \log n \leq |u_\lambda|$, in $\mathcal{O}(\alpha n)$ time. Further, we search maximal α -gapped repeats with shorter arms. Now, $|u_\lambda|$ is upper bounded by $2^{\log \log n + 1} \log n = 2(\log n)^2$, so $|u_\lambda u' u_\rho| \leq \ell_0$, for $\ell_0 = \alpha \cdot 2(\log n)^2 + 2(\log n)^2 = 2(\alpha + 1)(\log n)^2$. Such an α -gapped repeat $u_\lambda u' u_\rho$ is, thus, contained in (at least) one factor of length $2\ell_0$ of w , starting on a position of the form $1 + m\ell_0$ for $m \geq 0$. So, we take the factors $w[1 + m\ell_0, (m+2)\ell_0]$ of w , for $m \geq 0$, and apply for each such factor, separately, the same strategy as above to detect the maximal α -gapped repeats contained completely in each of them. The total time needed to do that is $\mathcal{O}\left(\alpha \ell_0 \frac{n}{\ell_0} + N_{\ell_0}\right) = \mathcal{O}(\alpha n)$, where N_{ℓ_0} is the number of repeats we find; moreover, we can easily ensure that a maximal repeat is not output twice (that is, ensure always that the gapped repeats we produce were not already contained in a previously processed interval). Hence, we find all maximal α -gapped repeats $u_\lambda u' u_\rho$ with $2^{\log \log(2\ell_0)} \log(2\ell_0) \leq |u|$. This means we find all the maximal α -gapped repeats with $|u| \geq 2^{\log \log(2\ell_0) + 1} \log(2\ell_0)$. Since $2^{\log \log(2\ell_0) + 1} \log(2\ell_0) \leq 16 \log n$ (for n large enough, as $\alpha \leq \log n$), we can apply Lemma 14 for gapped repeats with an arm-length smaller than $2^{\log \log(2\ell_0) + 1} \log(2\ell_0)$. ◀

Putting together the results of Lemmas 14 and 15 we get the following theorem.

► **Theorem 16.** *Given a word w and $\alpha \geq 1$, we can compute $\mathcal{G}_\alpha(w)$ in time $\mathcal{O}(\alpha n)$.*

By a completely similar approach we can compute $\mathcal{G}_\alpha^\Gamma(w)$, generalizing the algorithm of [13]. To this end, we construct *LCP*-structures for ww^Γ (allowing us to test efficiently whether a factor $w[i, j]^\Gamma$ occurs at some position in w). When we search the α -gapped palindromes u_λ, v, u_ρ (with $u_\rho \equiv u_\lambda^\Gamma$), we split again w in blocks and k -blocks, for each $k \leq \log |w|$, to check whether there exists such an u_λ, v, u_ρ with $2^k \leq |u_\lambda| \leq 2^{k+1}$. This search is conducted pretty much as in the case of repeats, only that now when we fix some factor y of u_ρ , we have to look for the occurrences of y^Γ in the factor of length $\mathcal{O}(\alpha |u_\rho|)$ preceding it; the *LCP*-structures for ww^Γ are useful for this, because, as explained above, they allow us to efficiently search the mirror images of factors of w inside w . Thus, given a word w and $\alpha \geq 1$, we can compute $\mathcal{G}_\alpha^\Gamma(w)$ in time $\mathcal{O}(\alpha n)$.

References

- 1 Hideo Bannai, Tomohiro I, Shunsuke Inenaga, Yuto Nakashima, Masayuki Takeda, and Kazuya Tsuruta. The Runs Theorem. *CoRR*, abs/1406.0263, 2014.
- 2 Gerth Stølting Brodal, Rune B. Lyngsø, Christian N. S. Pedersen, and Jens Stoye. Finding maximal pairs with bounded gap. In *Proc. 10th Annual Symposium on Combinatorial Pattern Matching*, volume 1645 of *LNCS*, pages 134–149. Springer, 1999.
- 3 Maxime Crochemore, Costas S. Iliopoulos, Marcin Kubica, Wojciech Rytter, and Tomasz Walen. Efficient algorithms for two extensions of LPF table: The power of suffix arrays. In *Proc. SOFSEM 2010*, volume 5901 of *LNCS*, pages 296–307, 2010.
- 4 Maxime Crochemore, Roman Kolpakov, and Gregory Kucherov. Optimal searching of gapped repeats in a word. *ArXiv e-prints 1309.4055*, 2015.
- 5 Maxime Crochemore and German Tischler. Computing longest previous non-overlapping factors. *Inf. Process. Lett.*, 111(6):291–295, February 2011.
- 6 Marius Dumitran and Florin Manea. Longest gapped repeats and palindromes. In *Proc. MFCS 2015*, volume 9234 of *LNCS*, pages 205–217. Springer, 2015.
- 7 Pawel Gawrychowski. Pattern matching in Lempel-Ziv compressed strings: Fast, simple, and deterministic. In *Proc. ESA*, volume 6942 of *LNCS*, pages 421–432, 2011.
- 8 Pawel Gawrychowski and Florin Manea. Longest α -gapped repeat and palindrome. In *Proc. FCT 2015*, volume 9210 of *LNCS*, pages 27–40. Springer, 2015.
- 9 Dan Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press, New York, NY, USA, 1997.
- 10 Juha Kärkkäinen, Peter Sanders, and Stefan Burkhardt. Linear work suffix array construction. *J. ACM*, 53:918–936, 2006.
- 11 Tomasz Kociumaka, Jakub Radoszewski, Wojciech Rytter, and Tomasz Walen. Efficient data structures for the factor periodicity problem. In *Proc. SPIRE*, volume 7608 of *LNCS*, pages 284–294, 2012.
- 12 Roman Kolpakov and Gregory Kucherov. Finding repeats with fixed gap. In *Proc. SPIRE*, pages 162–168, 2000.
- 13 Roman Kolpakov and Gregory Kucherov. Searching for gapped palindromes. *Theoretical Computer Science*, 410(51):5365 – 5373, 2009. Combinatorial Pattern Matching.
- 14 Roman Kolpakov, Mikhail Podolskiy, Mikhail Posypkin, and Nickolay Khrapov. Searching of gapped repeats and subrepetitions in a word. In *Proc. CPM*, volume 8486 of *LNCS*, pages 212–221, 2014.
- 15 Yuka Tanimura, Yuta Fujishige, Tomohiro I, Shunsuke Inenaga, Hideo Bannai, and Masayuki Takeda. A faster algorithm for computing maximal α -gapped repeats in a string. In *Proc. SPIRE 2015*, volume 9309 of *LNCS*, pages 124–136. Springer, 2015.
- 16 Peter van Emde Boas. Preserving order in a forest in less than logarithmic time. In *Proc. FOCS*, pages 75–84, 1975.

Appendix

6 A short recent history of this problem

The problem of searching for gapped repeats and palindromes in words is not so new (see [9, 2, 12]), and different solutions were proposed depending on the type of restrictions imposed on the gap. In [13], Kolpakov and Kucherov introduced the notion of α -gapped palindromes, and showed how to compute the set $\mathcal{G}_\alpha^I(w)$ of all maximal α -gapped palindromes in $\mathcal{O}(\alpha^2 n + |\mathcal{G}_\alpha^I(w)|)$ time for any input word w of length n over constant alphabets. In [14], Kolpakov et al. introduced the notion of α -gapped repeats, and showed that the number of maximal α -gapped repeats in a word w of length n is $\mathcal{O}(\alpha^2 n)$, and that this maximal set $\mathcal{G}_\alpha(w)$ can be computed in $\mathcal{O}(\alpha^2 n)$ time. They suggested two open problems concerning this:

- closing the gap between the upper bound $\mathcal{O}(\alpha^2 n)$ and the lower bound $\Omega(\alpha n)$ for the number of maximal α -gapped repeats, and
- developing a more efficient algorithm.

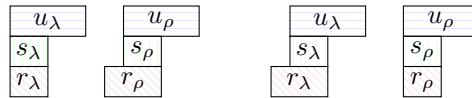
Their preliminary experiments suggested that the upper bound should be $\mathcal{O}(\alpha n)$.

In February 2015, at the Stringmasters meeting in Warsaw, the third author of this paper pointed that obtaining a bound on the number of α -gapped repeats seems to be an interesting open problem, and also conjectured that this bound is $\mathcal{O}(\alpha n)$. As a reaction, several connected papers appeared: [8] shows how to compute the longest α -gapped repeat/palindrome in $\mathcal{O}(\alpha n)$ time, [6] shows how to compute a series of data structures that can give the longest 2-gapped repeat/palindrome that starts at each position (and the results generalize easily to arbitrary α), [15] gives an $\mathcal{O}(\alpha n + |\mathcal{G}_\alpha(w)|)$ -time solution to find all maximal α -gapped repeats for an input word over constant alphabets.

Finally, in August 2015, the fourth author of this paper announced on the Stringmasters web-page¹ that the bound on the number of α -gapped repeats is indeed $\mathcal{O}(\alpha n)$; together with [15] this lead to an optimal algorithm for solving the problem of finding all α -gapped repeats in the particular case of constant alphabets. This announcement was followed by the paper [4] which basically confirmed the bound $\mathcal{O}(\alpha n)$ and also gave an algorithm computing efficiently all α -gapped maximal repeats for constant alphabets. Our paper concludes in big measure this line of research: we give concrete bounds on the number of α -gapped repeats and α -gapped palindromes, and, building on the approach from [8], we give optimal algorithms for finding them in the usual case of integer alphabets.

7 Gapped repeats

The following figure might ease the understanding the proof of Lemma 7 regarding periodic gapped repeats.



■ **Figure 5** The equation $b(u_\lambda) = b(r_\lambda)$ or $b(u_\rho) = b(r_\rho)$ must hold in the setting of Lemma 7. By the maximality property of runs, $e(r_\lambda) = e(s_\lambda)$ and $e(r_\rho) = e(s_\rho)$.

¹ http://stringmasters.mimuw.edu.pl/open_problems.html

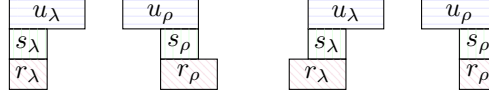
8 Gapped palindromes

► **Lemma 17.** *Given a word w , and two real numbers $\alpha > 1$ and $0 < \beta < 1$. Then $|\beta\mathcal{P}_\alpha^\Gamma(w)|$ is at most $2(\alpha + 1)E(w)/\beta$.*

Proof. Let $\sigma \in \beta\mathcal{P}_\alpha(w)$ (or $\sigma \in \beta\mathcal{P}_\alpha^\Gamma(w)$). The gapped palindrome σ consists of the triple $(u_\lambda, v, u_\rho) \in \beta\mathcal{P}_\alpha(w)$. By definition, the left arm u_λ has a periodic prefix s_λ of length at least $\beta|u_\lambda|$. Let r_λ denote the run that generates s_λ , i.e., $s_\lambda \subseteq r_\lambda$ and they both have the common shortest period p . By the definition of gapped palindromes, there is a right copy s_ρ of s_λ contained in u_ρ with

$$s_\rho = w[\mathbf{b}(u_\rho) + (\mathbf{e}(u_\lambda) - \mathbf{e}(s_\lambda)), \mathbf{b}(u_\rho) + (\mathbf{e}(u_\lambda) - \mathbf{e}(s_\lambda)) + |s_\lambda| - 1] \equiv s_\lambda^\top$$

Let r_ρ be a run generating s_ρ (it is possible that r_ρ and r_λ are identical). By definition, r_ρ has the same period p as r_λ . In the following, we will see that σ is uniquely determined by r_λ and the distance $d := \mathbf{b}(s_\rho) - \mathbf{e}(s_\lambda)$, if σ is a periodic gapped palindrome. We will fix r_λ and study how many maximal periodic gapped repeats (palindromes) can be generated by r_λ .



■ **Figure 6** The equation $\mathbf{b}(u_\lambda) = \mathbf{b}(r_\lambda)$ or $\mathbf{e}(u_\rho) = \mathbf{e}(r_\rho)$ must hold in the setting of Lemma 17. By the maximality property of runs, $\mathbf{e}(r_\lambda) = \mathbf{e}(s_\lambda)$ and $\mathbf{b}(r_\rho) = \mathbf{b}(s_\rho)$.

Since σ is maximal, $\mathbf{b}(u_\lambda) = \mathbf{b}(r_\lambda)$ or $\mathbf{e}(u_\rho) = \mathbf{e}(r_\rho)$ must hold; otherwise we could extend σ outwards. We analyze the case $\mathbf{b}(s_\lambda) = \mathbf{b}(r_\lambda)$, the other is treated exactly in the same way by symmetry. The gapped palindrome σ is identified by r_λ and d . We fix r_λ and count the number of possible values for d . Given two different periodic α -gapped palindromes with the distances d_1 and d_2 , the difference between d_1 and d_2 must be at least p , due to Lemma 2. It follows from $|u_\lambda| \leq |s_\lambda|/\beta$ that $d = |v| + 2(|u_\lambda| - |s_\lambda|) \leq |v| + 2|s_\lambda|/\beta$. Since σ is α -gapped, $|v| \leq (\alpha - 1)|u_\lambda| \leq (\alpha - 1)|s_\lambda|/\beta$, and hence, $1 \leq d \leq |s_\lambda|(\alpha + 1)/\beta$. Then the number of possible values for the distance d is bounded by $|s_\lambda|(\alpha + 1)/(\beta p) \leq |r_\lambda|(\alpha + 1)/(\beta p) = \exp(r_\lambda)(\alpha + 1)/\beta$. In total, the number of maximal α -gapped palindromes in this case is bounded by $(\alpha + 1)E(w)/\beta$ for the case $\mathbf{b}(u_\lambda) = \mathbf{b}(r_\lambda)$. Summing up we get the bound $2(\alpha + 1)E(w)/\beta$. ◀

► **Lemma 18.** *Given a word w , and two real numbers $\alpha > 1$ and $6/7 \leq \beta < 1$. The points mapped by two different maximal gapped palindromes in $\beta\mathcal{P}_\alpha^\Gamma(w)$ cannot $\frac{1-\beta}{\alpha}$ -cover the same point.*

Proof. Let $\sigma = u_\lambda, v, u_\rho$ and $\bar{\sigma} = \bar{u}_\lambda, \bar{v}, \bar{u}_\rho$ be two different gapped palindromes in $\beta\mathcal{P}_\alpha^\Gamma(w)$. Set $u := |u_\lambda| = |u_\rho|$, $\bar{u} := |\bar{u}_\lambda| = |\bar{u}_\rho|$, $g := |v|$ and $\bar{g} := |\bar{v}|$. We map the maximal gapped palindromes σ and $\bar{\sigma}$ to the points $(\mathbf{e}(u_\lambda), g)$ and $(\mathbf{e}(\bar{u}_\lambda), \bar{g})$, respectively. Assume, for the sake of contradiction, that both points $\frac{1-\beta}{\alpha}$ -cover the same point (x, y) .

Let $z := |\mathbf{e}(u_\lambda) - \mathbf{e}(\bar{u}_\lambda)|$ be the difference of the endings of both left arms, and $s_\lambda := w[\mathbf{b}(u_\lambda), \mathbf{e}(u_\lambda)] \cap [\mathbf{b}(\bar{u}_\lambda), \mathbf{e}(\bar{u}_\lambda)]$ be the overlap of u_λ and \bar{u}_λ . Let $s = |s_\lambda|$, and let s_ρ (resp. \bar{s}_ρ) be the reversed copy of s_λ based on σ (resp. $\bar{\sigma}$).

Sub-Claim: The overlap s_λ is not empty, and $s_\rho \neq \bar{s}_\rho$

Sub-Proof. Assume for this sub-proof that $\mathbf{e}(u_\lambda) < \mathbf{e}(\bar{u}_\lambda)$ (otherwise exchange σ with $\bar{\sigma}$, or yield the contradiction $\sigma = \bar{\sigma}$). By combining the $(1 - \beta)/\alpha$ -cover property with the

fact that $\bar{\sigma}$ is α -gapped, we yield $\mathbf{e}(\bar{u}_\lambda) - \bar{u} \leq \mathbf{e}(\bar{u}_\lambda) - \bar{g}(1 - \beta)/\alpha \leq x \leq \mathbf{e}(u_\lambda) < \mathbf{e}(\bar{u}_\lambda)$. So the subword $w[\mathbf{e}(u_\lambda)]$ is contained in \bar{u}_λ . If $s_\rho = \bar{s}_\rho$, then we get a contradiction to the maximality of σ : By the above inequality, $w[\mathbf{e}(u_\lambda) + 1]$ is contained in \bar{u}_λ , too. Since $\bar{\sigma}$ is a gapped palindrome, the character $w[\mathbf{e}(u_\lambda) + 1]$ occurs in \bar{u}_ρ , exactly at $w[\mathbf{b}(u_\rho) - 1]$. \blacktriangleleft

Without loss of generality let $g \leq \bar{g}$. Then

$$\bar{g} - \frac{\bar{g}(1 - \beta)}{\alpha} \leq y \leq g \leq \bar{g}. \quad (6)$$

So the difference of both gaps is

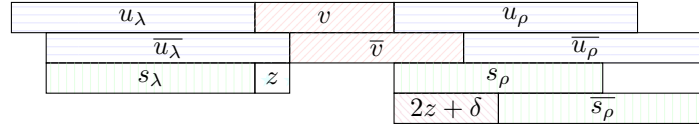
$$0 \leq \delta := \bar{g} - g \leq \bar{g}(1 - \beta)/\alpha \leq \bar{u}(1 - \beta). \quad (7)$$

By case analysis, we show that s_λ is periodic, which leads to the contradiction that σ or $\bar{\sigma}$ are in $\beta\mathcal{P}_\alpha^\Gamma(w)$.

1. Case $\mathbf{e}(u_\lambda) \leq \mathbf{e}(\bar{u}_\lambda)$. Since $\mathbf{e}(\bar{u}_\lambda) - \bar{g}(1 - \beta)/\alpha \leq x \leq \mathbf{e}(u_\lambda) \leq \mathbf{e}(\bar{u}_\lambda)$,

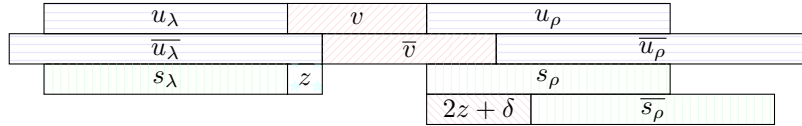
$$z = \mathbf{e}(\bar{u}_\lambda) - \mathbf{e}(u_\lambda) \leq \bar{g}(1 - \beta)/\alpha \leq \bar{u}(1 - \beta). \quad (8)$$

Since s_λ is a prefix of \bar{u}_λ and a suffix of u_λ , the reverse copy s_ρ is a suffix of \bar{u}_ρ and a prefix of u_ρ . The starting positions of both right copies \bar{s}_ρ and s_ρ differ by $\mathbf{b}(\bar{s}_\rho) - \mathbf{b}(s_\rho) = 2z + \delta > 0$. By Equations (7) and (8), we get $2z + \delta \leq 3\bar{g}(1 - \beta)/\alpha \leq 3\bar{u}(1 - \beta)$.



■ **Figure 7** Sub-Case 1a

1a. Sub-Case $\mathbf{b}(u_\lambda) \leq \mathbf{b}(\bar{u}_\lambda)$. By Equation (8), we get $s = \bar{u} - z \geq \bar{u}\beta$. It follows from $6/7 \leq \beta < 1$ that $s/(2z + \delta) \geq \bar{u}\beta/3\bar{u}(1 - \beta) = \beta/(3(1 - \beta)) \geq 2$, which means that s_ρ and \bar{s}_ρ overlap by at least half of their common length, and s_λ is periodic. Since s_λ is a prefix of \bar{u}_λ of length $s \geq \bar{u}\beta$, $\bar{\sigma}$ is in $\beta\mathcal{P}_\alpha^\Gamma(w)$, a contradiction.



■ **Figure 8** Sub-Case 1b

1b. Sub-Case $\mathbf{b}(u_\lambda) > \mathbf{b}(\bar{u}_\lambda)$. We conclude that $s_\lambda = u_\lambda$. By Equation (6),

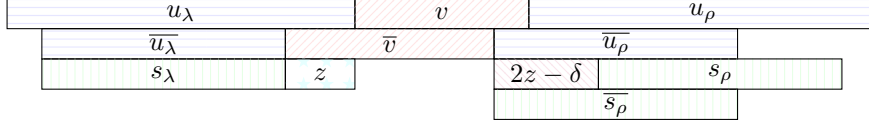
$$u \geq g/\alpha \geq \frac{\bar{g}}{\alpha}(1 - \frac{1 - \beta}{\alpha}) \geq \bar{g}\beta/\alpha. \quad (9)$$

It follows from $6/7 \leq \beta < 1$ that $s/(2z + \delta) \geq \bar{g}\alpha\beta/(3\alpha\bar{g}(1 - \beta)) = \beta/(3(1 - \beta)) \geq 2$, which means that $s_\lambda = u_\lambda$ is periodic. Hence σ is in $\beta\mathcal{P}_\alpha^\Gamma(w)$, a contradiction.

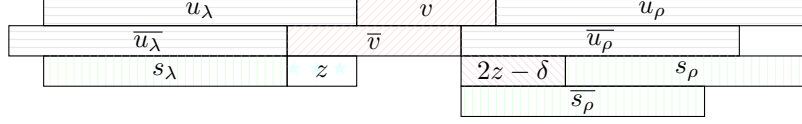
2. Case $\mathbf{e}(u_\lambda) > \mathbf{e}(\bar{u}_\lambda)$. Since $\mathbf{e}(u_\lambda) - g(1 - \beta)/\alpha \leq x \leq \mathbf{e}(\bar{u}_\lambda) \leq \mathbf{e}(u_\lambda)$,

$$z = \mathbf{e}(u_\lambda) - \mathbf{e}(\bar{u}_\lambda) \leq g(1 - \beta)/\alpha \leq \bar{g}(1 - \beta)/\alpha \leq \bar{u}(1 - \beta). \quad (10)$$

The starting positions of both right copies differ by $|\mathbf{b}(s_\rho) - \mathbf{b}(\bar{s}_\rho)| = |2z - \delta|$. Since $2z - \delta \leq \max(\delta, 2z)$, we get $|2z - \delta| \leq 2\bar{g}(1 - \beta)/\alpha \leq 2\bar{u}(1 - \beta)$ by Equations (7) and (10).



■ **Figure 9** Sub-Case 2a



■ **Figure 10** Sub-Case 2b

2a. Sub-Case $b(u_\lambda) \leq b(\overline{u_\lambda})$. We conclude that $s_\lambda = \overline{u_\lambda}$. It follows from $6/7 \leq \beta < 1$ that $s/|2z - \delta| \geq \overline{u}/(2\overline{u}(1 - \beta)) = 1/(2(1 - \beta)) \geq 7/2 > 2$, which means that $s_\lambda = \overline{u_\lambda}$ is periodic. Hence $\overline{\sigma}$ is in $\beta\mathcal{P}_\alpha^\Gamma(w)$, a contradiction.

2b. Sub-Case $b(u_\lambda) > b(\overline{u_\lambda})$. By Equation (10), we get $z \leq u(1 - \beta)$ and thus $s = u - z \geq \beta u$. It follows from Equations (7) and (9), and $2(\sqrt{2} - 1) < 6/7 \leq \beta < 1$ that $s/|2z - \delta| \geq \beta u/(2\overline{g}(1 - \beta)/\alpha) \geq \overline{g}\beta^2/(2\overline{g}(1 - \beta)) = \beta^2/(2(1 - \beta)) > 2$, which means that s_λ is periodic. Since s_λ is a prefix of u_λ of length $s \geq u\beta$, σ is in $\beta\mathcal{P}_\alpha^\Gamma(w)$, a contradiction. ◀

The next lemma follows immediately from Lemmas 6 and 18.

► **Lemma 19.** For $\alpha > 1$, $6/7 \leq \beta < 1$ and a word w of length n , $|\overline{\beta\mathcal{P}_\alpha^\Gamma(w)}| < 3\alpha n/(1 - \beta)$.

► **Theorem 20.** For $\alpha > 1$ and a word w of length n , $|\mathcal{G}_\alpha^\Gamma(w)| < 28\alpha n + 7n$.

Proof. By Lemmas 7 and 19, $|\mathcal{G}_\alpha^\Gamma(w)| = |\beta\mathcal{P}_\alpha^\Gamma(w)| + |\overline{\beta\mathcal{P}_\alpha^\Gamma(w)}| < 2(\alpha + 1)E(w)/\beta + 3\alpha n/(1 - \beta)$ for every $6/7 \leq \beta < 1$. Applying Lemma 1, the term is upper bounded by $6(\alpha + 1)n/\beta + 3\alpha n/(1 - \beta)$. This number is minimal when $\beta = 6/7$, yielding the bound $28\alpha n + 7n$. ◀

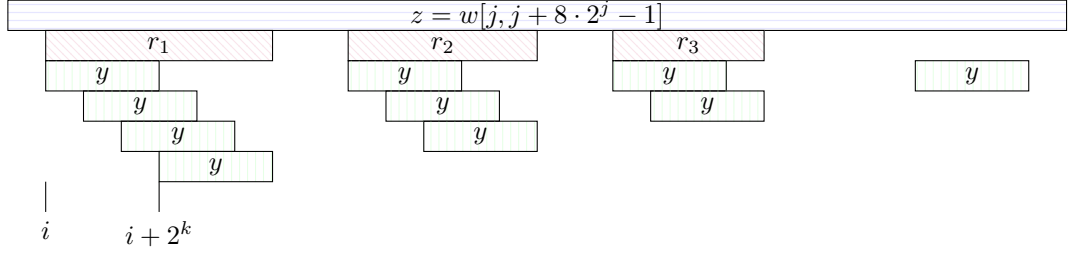
Auxiliary algorithmic results

Proof of Lemma 11.

Proof. Once we produce *LCP* data structures for w , we just have to compute the longest common prefix of $w[i, n]$ and $w[i + p, n]$. If this prefix is $w[i + p, \ell]$, then $w[i, \ell]$ is the longest p -periodic factor starting on position i . ◀

Proof of Lemma 12

Proof. We construct the dictionary of basic factors of a word of length n in $\mathcal{O}(n \log n)$ time and reorganise it such that for each basic factor we have an array with all its occurrences, ordered by their starting positions. For each such array we construct data structures that allow predecessor/successor search in $\mathcal{O}(\log \log n)$ time (see, e.g., [16]). When we have to return the occurrences of $y = w[i, i + 2^k - 1]$ in $z = w[j, j + c2^k - 1]$, we search in the basic factors-array corresponding to $w[i, i + 2^k - 1]$ the successor of j and, respectively, the predecessor of $j + c2^k - 1$ and then return a succinct representation of the occurrences of $w[i, i + 2^k - 1]$ between these two values. This representation can be obtained in $\mathcal{O}(c)$ time.



■ **Figure 11** Occurrences of the basic factor $y = w[i, i + 2^k - 1]$ in $z = w[j, j + 8 \cdot 2^j - 1]$. The overlapping occurrences are part of runs, and they can be returned as such. The representation of the occurrences of y in z will return 4 elements: 3 runs and one separate occurrence.

We just have to detect those occurrences that form a run; this can be done with a constant number of *LCP* queries. Indeed, for two consecutive occurrences, we compute the length of their overlap, which gives us a period of $w[i, i + 2^k - 1]$. Then we look in w to see how long the run with this period can be extended to the right, which gives us the number of occurrences of $w[i, i + 2^k - 1]$ in that run. As their starting positions form an arithmetic progression, we can represent them compactly. So, we return the representation of the occurrences of $w[i, i + 2^k - 1]$ from this run, and then move directly to the first occurrence of $w[i, i + 2^k - 1]$ appearing after this run and still in the desired range; as there are at most $\mathcal{O}(c)$ runs and separate occurrences of the given basic factor that are in z , the conclusion follows. ◀

Proof of Lemma 13.

In [8] the following result was shown. Lemma 13 follows directly.

► **Lemma 21.** *Given a word v , $|v| = \alpha \log n$, we can process v in time $\mathcal{O}(\alpha \log n)$ time such that given any basic factor $y = v[j \cdot 2^k + 1, (j + 1)2^k]$ with $j, k \geq 0$ and $j2^k + 1 > (\alpha - 1) \log n$, we can construct in $\mathcal{O}(\alpha)$ time $\mathcal{O}(\alpha)$ bit-sets, each storing $\mathcal{O}(\log n)$ bits, characterizing all the occurrences of y in v .*

The following lemma is useful for the proof. More precisely, one can also construct data structures that allow us fast identification of the suffixes of a word that start with a given basic factor.

► **Lemma 22** ([7]). *A word w of length n can be processed in $\mathcal{O}(n)$ time such that given any basic factor $w[i, i + 2^k - 1]$ with $k \geq 0$, we can retrieve in $\mathcal{O}(1)$ time the range of the suffix array of w of suffixes starting with $w[i, i + 2^k - 1]$. Equivalently, we can find the node of the suffix tree of w closest to the root, such that the label of the path from the root to that node has the prefix $w[i, i + 2^k - 1]$.*

The proof of Lemma 21 follows.

Proof. We first show how the proof works for $\alpha = 1$.

We first build the suffix tree for v in $\mathcal{O}(\log n)$ time. We further process this suffix tree such that we can find in constant time, for each factor $v[j \cdot 2^k + 1, (j + 1)2^k]$, the node u of the suffix tree which is closest to the root with the property that the label of the path from the root to u starts with $v[j \cdot 2^k + 1, (j + 1)2^k]$. According to Lemma 22, this can be done in linear time.

Now, we augment the suffix tree in such a manner that for each node we store an additional bit-set, indicating the positions of v where the word labelling the path from the root to the respective node occurs. Each of these bit-sets, of size $\mathcal{O}(\log n)$ bits, can be stored in constant space; indeed, each $\log n$ block of bits can be seen in fact as a number between 1 and n so we only need to store a constant number of numbers smaller than n ; in our model, each such number fits in a memory word. Computing the bit-sets can be done in a bottom up manner in linear time: for a node, we need to make the union of the bit-sets of its children, and this can be done by doing a bitwise **or** operation between all the bit-sets corresponding to the children. So, now, checking the bit-set associated to the lowest node of the suffix tree such that the label of the path from the root to that node starts with $v[j \cdot 2^k + 1, (j + 1)2^k]$ we can immediately output a representation of this factor's occurrences in v .

This concludes the proof for $\alpha = 1$.

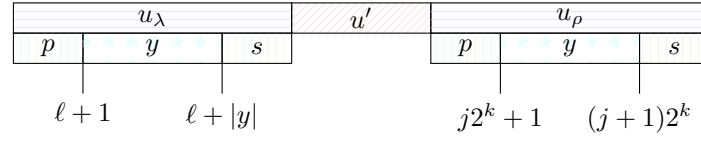
For $\alpha > 1$, we just have to repeat the algorithm in the previous proof for the words $v[i \log n + 1, (i + 2) \log n]v[(\alpha - 1) \log n + 1, \alpha \log n]$, for $0 \leq i \leq \alpha - 2$, which allows us to find all the occurrences of the basic factors of $v[(\alpha - 1) \log n + 1, \alpha \log n]$ in v . The time is clearly $\mathcal{O}(\alpha)$. ◀

► **Remark.** By this lemma, given a word v , $|v| = \alpha \log n$, and a basic factor $y = v[j \cdot 2^k + 1, (j + 1)2^k]$, with $j, k \geq 0$ and $j2^k + 1 > (\alpha - 1) \log n$, we can produce $\mathcal{O}(\alpha)$ bit-sets, each containing exactly $\mathcal{O}(\log n)$ bits, characterising all the occurrences of y in v . Let us also assume that we have access to all values $\log x$ with $x \leq n$ (which can be ensured by a $\mathcal{O}(n)$ preprocessing). Now, using the bit-sets encoding the occurrences of y in v and given a factor z of v , $|z| = c|y|$ for some $c \geq 1$, we can obtain in $\mathcal{O}(c)$ time the occurrences of y in z : the positions (at most c) where y occurs outside a run and/or at most c runs containing the occurrences of y . Indeed, the main idea is to select by bitwise operations on the bit-sets encoding the factors of v that overlap z the positions where y occurs (so the positions with an 1). For each two consecutive such occurrences of y we detect whether they are part of a run in v (by *LCP*-queries on v) and then skip over all the occurrences of y from that run (and the corresponding parts of the bit-sets) before looking again for the 1-bits in the bit-sets.

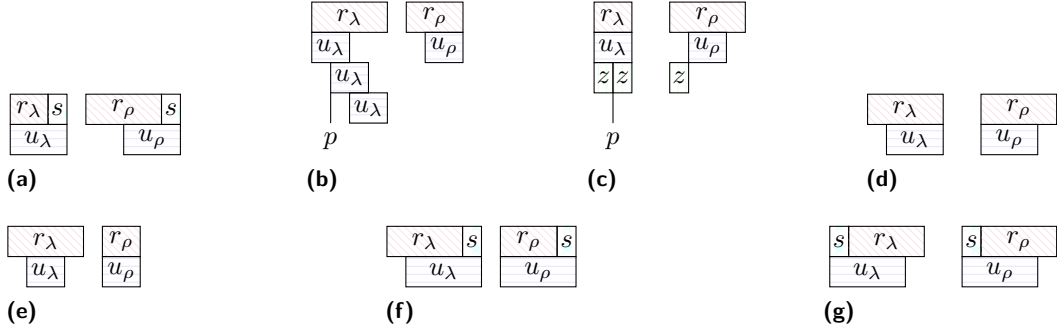
Proof. When given an additional subrange z of v , we have to select from the non-null bits of the bit-sets corresponding to the factors $v[j \log n + 1, (j + 2) \log n]$ of v those corresponding to the range z . This can be easily done with bitwise operations in $\mathcal{O}(c)$ time. Indeed, when looking at such a bit-set, we first select the most significant 1-bit (it occurs on the position given by $\log m$, where m is the integer value of the current bit-set), it gives an occurrence of y . Then we turn this bit into 0, and select the next most-significant 1, which again gives an occurrence of y . If these two occurrences are part of a run, we detect the ending position of the run by a *LCP* query on v , and continue finding the y 's after that position. Otherwise, we repeat the procedure (the second y detected able will be now detected as the first y). ◀

Missing cases from the proof of Lemma 14.

Assume u_ρ starts on the first position of r_ρ . If u_ρ ends on a position inside r_ρ , other than its last position, then u_λ should end on the last position of r_λ (otherwise, both arms could be extended to the right). This means that we know exactly the gap between the two arms of the α -gapped repeat that we want to construct, as well as the fact that the arms are p -periodic. We consider, again, the longest p -periodic suffix of r_λ which is a prefix of r_ρ , and see whether the two occurrences of this factor determine a maximal α -gapped repeat. If u_ρ ends on the last position of r_ρ then we know the exact arm of the α -gapped palindrome we are looking for (that is, if it also fulfils the length conditions). We can proceed just like



■ **Figure 12** Fixing x_m in the proof of Theorem 14, we try to spot gapped repeats whose arms contain a certain basic factor. If we can extend this gapped repeat to a maximal gapped repeat, we output it.



■ **Figure 13** Catching gapped repeats with periodicity is done by case analysis in the proof of Theorem 14. Each case is depicted in order (from left to right, top to bottom)

in the case analyzed before, when we knew that $u_\lambda = r_\lambda$, only that in this case we have to determine the positions where u_λ may start instead of those where u_ρ started. Finally, if u_ρ ends on a position to the right of r_ρ , then u_λ should also end after r_λ and the suffix of u_λ occurring after r_λ should be equal to the suffix of u_ρ occurring after r_ρ . We determine this suffix by a longest common prefix query on x_m , and then we obtain exactly the arms of the repeat. We can check whether it is a valid maximal α -gapped repeat, and, if so, return it.

The last case is when u_ρ starts on a position to the left of r_ρ . Then u_λ starts on a position occurring before the first position of r_λ , the prefix of u_ρ occurring before the beginning of r_ρ equals the prefix of u_λ occurring before r_λ , and they can be determined in constant time. Thus we know the starting points of both arms of the repeat, and we can determine them exactly by a longest common prefix query. We just have to check whether the arms we obtained form a valid maximal α -gapped repeat.